Approximation Transducers and Trees: A Technique for Combining Rough and Crisp Knowledge

Patrick Doherty[1], Witold Łukaszewicz[2], Andrzej Skowron[3], and Andrzej Szałas[2]

[1] Department of Computer and Information Science, Linköping University, SE-581 83 Linköping, Sweden, `e-mail: patdo@ida.liu.se`
[2] The College of Economy and Computer Science, Wyzwolenia 30, 10-106 Olsztyn, Poland, `e-mail: {witlu,andsz}@ida.liu.se`
[3] Institute of Mathematics, Warsaw University, Banacha 2, 02-097 Warsaw, Poland, `e-mail: skowron@mimuw.edu.pl`

Summary. This chapter proposes a framework for specifying, constructing, and managing a particular class of approximate knowledge structures for use with intelligent artifacts ranging from simpler devices such as personal digital assistants to more complex ones such as unmanned aerial vehicles. The notion of an *approximation transducer* is introduced which takes approximate relations as input, and generates a (possibly more abstract) approximate relation as output by combining the approximate input relations with a crisp local logical theory which represents dependencies between the input and output relations. Approximation transducers can be combined to produce *approximation trees* which represent complex approximate knowledge structures characterized by the properties of elaboration tolerance, groundedness in the application domain, modularity, and context dependency. Approximation trees are grounded through the use of primitive concepts generated with supervised learning techniques. Changes in definitions of primitive concepts or in the local logical theories used by transducers result in changes in the knowledge stored in approximation trees by increasing or decreasing precision in the knowledge in a qualitative manner. Intuitions and techniques from rough set theory are used to define approximate relations where each has an upper and lower approximation. The constituent components in a rough set have correspondences in a logical language used to relate crisp and approximate knowledge. The inference mechanism associated with the use of approximation trees is based on a generalization of deductive databases that we call *rough relational databases*. Approximation trees and queries to them are characterized in terms of rough relational databases and queries to them. By placing certain syntactic restrictions on the local theories used in transducers, the computational processes used in the query/answering and generation mechanism for approximation trees remain in PTIME.

1 Introduction and Background

In this introductory section, we will set the context for the knowledge representation framework pursued in this chapter. We begin with a discussion of intelligent artifacts and society of agent frameworks [7]. We proceed to a discussion of knowledge representation components for agents and consider the need for self-adapting knowledge representation structures and concept acquisition techniques. The core idea pursued in this chapter is to propose a framework for the specification, construction and management of approximate knowledge structures for intelligent artifacts. The specific structures used are called approximation transducers and approximation trees. The specific implementation framework used is based on a generalization of deductive database technology. We describe the intuitions and basic ideas behind these concepts. We then conclude the introductory section with a brief description of the experimental platform from which these ideas arose and from which we plan to continue additional experimentation with the framework. The experimental platform is a deliberative/reactive software architecture for an unmanned aerial vehicle under development in the WITAS Unmanned Aerial Vehicle Project at Linköping University, Sweden[1].

1.1 Intelligent Artifacts and Agents

The use of intelligent artifacts both at the workplace and in the home is becoming increasingly more pervasive due to a number of factors which include the accessibility of the Internet/World-Wide-Web to the broad masses, the drop in price and increase in capacity of computer processors and memory, and the integration of computer technology with telecommunications. Intelligent artifacts are man-made physical systems containing computational equipment and software that provide them with capabilities for receiving and comprehending sensory data, for reasoning, and for performing rational action in their environment. The spectrum of capabilities and the sophistication of an artifact's ability to interface to its environment and reason about it varies with the type of artifact, its intended tasks, the complexity of the environment in which it is embedded, and its ability to adapt its models of the environment at different levels of knowledge abstraction. Representative examples of intelligent artifacts ranging from less to more complex would be mobile telephones, mobile telephones with Blue Tooth wireless technology[1], personal digital assistants (PDAs), collections of distributed communicating artifacts which serve as components of smart homes, mobile robots, unmanned aerial vehicles, and many more.

One unifying conceptual framework that can be used to view these increasingly more complex integrated computer systems is as societies of agents (virtually and/or physically embedded in their respective environments) with the capacity to acquire information about their environments, structure the information and interpret it as knowledge, and use this knowledge in a rational manner to enhance goal-directed behavior which is used to achieve tasks and to function robustly in their dynamic and complex environments.

[1] BLUETOOTH is a trademark owned by Telefonaktiebolaget L M Ericsson, Sweden.

1.2 Knowledge Representation

An essential component in agent architectures is the agent's knowledge representation component which includes a variety of knowledge and data repositories with associated inference mechanisms. The knowledge representation component is used by the agent to provide it with models of its embedding environment and of its own and other agent capabilities in addition to reasoning efficiently about them. It is becoming increasingly important to move away from the notion of a single knowledge representation mechanism with one knowledge source and inference method to multiple forms of knowledge representation with several inference methods. This viewpoint introduces an interesting set of complex research issues related to the merging of knowledge from disparate sources and the use of adjudication or conflict resolution policies to provide coherence of knowledge sources.

Due to the embedded nature of these agent societies in complex dynamic environments, it is also becoming increasingly important to take seriously the gap between access to low-level sensory data and its fusion and integration with more qualitative knowledge structures. These signal-to-symbol transformations should be viewed as an on-going process with a great deal of feedback between the levels of processing. In addition, because the embedding environments are often as complex and dynamic as those faced by humans, the knowledge representations which are used as models of the environment must necessarily be partial, elaboration tolerant and approximate in nature.

Self-Adaptive Knowledge Structures A long term goal with this research is to develop a framework for the specification, implementation and management of self-adaptive knowledge structures containing both quantitative and qualitative components, where the knowledge structures are grounded in the embedding environments in which they are used. There has been very little work in traditional knowledge representation with the dynamics and management of knowledge structures. Some related work would include the development of belief revision and truth maintenance systems in addition to the notion of *elaboration tolerant* knowledge representation and the use of *contexts* as first-class objects introduced by McCarthy [6]. In these cases, the view pertaining to properties and relations is still quite traditional with little emphasis on the approximate and contextual character of knowledge. The assumed granularity of the primitive components of these knowledge structures, in which these theories and techniques are grounded, is still that of classical properties and relations in a formal logical context. We will assume a finer granularity as a basis for concept acquisition, grounding and knowledge structure design which is the result of using intuitions from rough set theory.

Approximate Concept Acquisition and Management One important component related to the ontology used by agents is the acquisition, integration, and elaboration tolerant update of concepts. Here we will interpret the notion of *concept* in a broad sense. It will include both properties of the world and things in it and relations between them. The concepts and relations can be epistemic in nature and we will assume the agent architectures contain syntactic components which correlate with these concepts and relations. The symbol grounding problem, that of associating symbols with individuals, percepts and concepts and managing these associations, will be discussed and the knowledge representation technique proposed will provide an interesting form of grounding, which we hope can contribute to an eventual solution to this important, complex, and as yet unsolved problem. The symbol grounding problem includes not only the association of symbols with concepts, but also ongoing concept acquisition and modification during the life of an agent. This aspect of the problem will involve the use of concept learning techniques and their direct integration in the knowledge structures generated.

To do this, we will assume that certain concepts which we call *primitive concepts* have been acquired through a learning process where learning samples are provided from sensor data and approximations of concepts are induced from the data. One particularly interesting approach to this is the use of rough set based supervised learning techniques. It is important to emphasize that the induced concepts are approximate in nature and fluid in the sense that additional learning may modify the concept. In other words, concepts are inherently contextual and subject to elaboration and change in a number of ways. Primitive concepts may change as new sensor data is acquired and fused with existing data through diverse processes associated with particular sensory platforms. At some point, constraints associated with other more abstract concepts having dependencies with primitive concepts may influence the definition of the primitive concept.

As an example of these ideas, take a situation involving an unmanned aerial vehicle operating over a road and traffic environment. In this case, the meaning of concepts such as *fast* or *slow*, *small* or *large vehicle*, *near*, *far*, or *between*, will have a meaning different from that in another application with other temporal and spatial constraints.

Assuming these primitive concepts as given and that they are continually re-grounded in changes in operational environment via additional learning or sensor fusion, we would then like to use these primitive concepts as the *ur*-elements in our knowledge representation structures. Since these *ur*-elements are inherently approximate, contextual and elaboration tolerant in nature, any knowledge structure containing these concepts should also inherit or be influenced by these characteristics. In fact, there are even more primitive *ur*-elements in the system we envision which can be used to define the primitive concepts themselves if a specific concept learning policy based on rough sets is used. These are the elementary sets used in rough set theory to define contextual approximations to sets.

1.3 Approximation Transducers and Trees

In the philosophical literature, W. V. O. Quine [11] has used the phrase *web of belief* to capture the intricate and complex dependencies and structures which make up human beliefs. In this chapter and in a companion chapter in this book [2], we lay the ground work for what might properly be called *webs of approximate knowledge*. In fact, a better way to view this idea is as starting with *webs of imprecise knowledge* and gradually incrementing these initial webs with additional approximate and sometimes crisp facts and knowledge. Through this process, a number of concepts, relations and dependencies between them become less imprecise and more approximate in nature. There is a continual elastic process where the precision in meaning of concepts is continually modified in a change-tolerant manner. Approximate definitions of concepts will be the rule rather than the exception even though crisp definitions of concepts are a special case included in the framework.

Specifically, webs of approximate knowledge will be constructed from primitive concepts together with what we will call *approximation transducers* in a recursive manner. An approximation transducer provides an approximate definition of one or more output concepts in terms of a set of input concepts and consists of three components:

1. An input consisting of one or more approximate concepts, some of which might be primitive.
2. An output consisting of one or more new and possibly more abstract concepts defined partly in terms of the input concepts.
3. A local logical theory specifying constraints or dependencies between the input concepts and the output concepts. The theory may also refer to other concepts not expressed in the input.

The local logical theory specifies dependencies or constraints an expert for the application domain would be able to specify. Generally the form of the constraints would be in terms of some necessary and some sufficient conditions for the output concept. The local theory is viewed as a set of *crisp* logical constraints specified in the language of first-order logic. The local theory serves as a logical template. During the generation of the approximate concept output by the transducer, the crisp relations mentioned in the local theory are substituted with the actual approximate definitions of the input. Either lower or upper approximations of the input concepts may be used in the substitution. The resulting output specifies the output concept in terms of newly generated lower and upper approximations. The resulting output relation may then be used as input to other transducers creating what we call *approximation trees*. The resulting tree represents a web of approximate knowledge capturing intricate and complex dependencies among an agent's conceptual vocabulary.

As an example of a transducer that might be used in the unmanned aerial vehicle domain, we can imagine defining a transducer for the approximate concept of two vehicles being *connected* in terms of *visible connection*, *small distance*, and *equal speed*. The latter three input concepts could be generated from supervised learning techniques where the data is acquired from a library of videos previously collected by the UAV on earlier traffic monitoring missions. As part of the local logical theory, an example of a constraint might state that "if two vehicles are *visibly connected*, are at a *small distance* from each other and have *equal speed* then they are *connected*".

Observe that the resulting approximation trees are highly fluid, approximate, and elaboration tolerant. Changes in the definition of primitive concepts will trickle through the trees via the dependencies and connections, modifying some of the other concept definitions. Changes to the local theories anywhere in the tree will modify those parts of the tree related to the respective output concepts for the local theories. This is a form of elaboration tolerance. These structures are approximate in three respects:

1. The primitive concepts themselves are approximate and generated through learning techniques. In the case of rough learning techniques, they consist of upper and lower approximations induced from the sample data.
2. The output concepts inherit or are influenced by the approximate aspects of the concepts input to their respective transducers.
3. The output concepts also inherit the incompletely specified sufficient and necessary conditions in the local logical theory specified in part with the input concepts.

It is important to point out that the transducers represent a technique for combining both approximate and crisp knowledge. The flow of knowledge through a transducer transforms the output concept from a less precise towards a more approximate definition. The definition can continually be elaborated upon both directly by modifying the local theory and indirectly via the modification of concept definitions on which it is recursively dependent or through retraining of the primitive concepts through various learning techniques.

1.4 The WITAS UAV Experimental Platform

The ultimate goal of the research described in this chapter is to use it as a basis for specifying, constructing and managing a particular class of approximate knowledge structures in intelligent artifacts. In current research, the particular artifact we use as an experimental platform is an unmanned aerial vehicle flying over operational environments populated by traffic. In such a scenario, knowledge about both the environment below and the unmanned aerial vehicle agent's own epistemic state must be acquired in a timely manner in order for the knowledge to be of use to the agent while achieving its goals. Consequently, the result must provide for an efficient implementation of both the knowledge structures themselves and the inference mechanisms used to query these structures for information.

WITAS (pronounced vee-tas) is an acronym for the Wallenberg Information Technology and Autonomous Systems Laboratory at Linköping University. UAV is an acronym for Unmanned Aerial Vehicles. The WITAS UAV Project is a long term project with the goal of designing, specifying and implementing the IT subsystem for an intelligent autonomous aircraft and embedding it in an actual platform [1]. In our case, we are using a Yamaha RMAX VTOL (vertical take-off and landing system) developed by the Yamaha Motor Company Ltd. An important part of the project is in identifying core functionalities required for the successful development of such systems and doing basic research in the areas identified. The topic of this paper is one such core functionality: approximate knowledge structures and their associated inference mechanisms.

The project encompasses the design of a command and control system for a UAV and its integration in a suitable deliberative/reactive architecture; the design of high-level cognitive tasks, intermediate reactive behaviors, low-level control-based behaviors and their integration with each other; the integration of sensory capabilities with the command and control architecture, in particular the use of an active vision system; the development of hybrid, mode-based low-level control systems to supervise and schedule control behaviors; the signal-to-symbol conversions from sensory data to qualitative structures used in mediating choice of actions and synthesizing plans to attain operational mission goals; and the development of the systems architecture for the physical UAV platform.

In addition the project also encompasses the design and development of the necessary tools and research infrastructure required to achieve the goals of the project. This would include the development of model-based distributed simulation tools and languages used in the concurrent engineering required to move incrementally from software emulation and simulation to the actual hardware components used in the final product.

The intended operational environment is over widely varying geographical terrain with traffic networks and vehicle interaction of varying degrees of density. Possible applications are emergency services assistance, monitoring and surveillance, use of a UAV as a mobile sensory platform in an integrated real-time traffic control system and photogrammetry applications.

The UAV experimental platform offers an ideal environment for experimentation with the knowledge representation framework we propose, because the system architecture is rich with different types of knowledge representation structures, the operational environment is quite complex and dynamic, and signal-to-symbol transformations of data are an integral part of the architecture. In addition, much of the knowledge acquired by the UAV will be necessarily approximate in nature. In several of the sections in this chapter, we will use examples from this application domain to describe and motivate some of our techniques.

1.5 Generalized Deductive Databases

Due to the pragmatic constraints associated with the deployment and execution of systems such as the WITAS UAV platform, we will develop these ideas in the context of deductive database systems. Deductive database systems offer a reasonable compromise between expressiveness of the language used to model aspects of the embedding environment and the efficiency of the inference mechanisms required in the soft and sometimes hard real-time contexts in which the UAV operates. A deductive database concentric view of data and knowledge flow in a generic artificial intelligence application contains a number of components as depicted in Fig. 1.

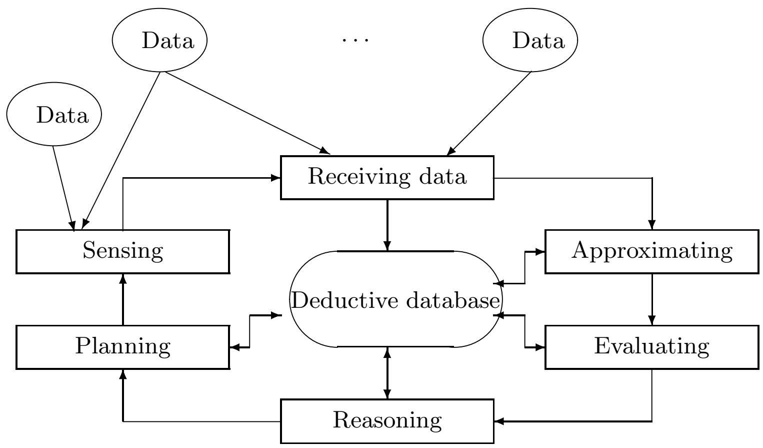

Fig. 1. Deductive Database Concentric View of Data and knowledge flow

In the UAV architecture, there are in fact a number of databases, or knowledge and data repositories. The dynamic object repository is a soft real-time database used to store pre-processed sensory data from the sensor platform and includes information about objects identified as moving in the traffic scenarios observed by the UAV when flying over particular road systems. In addition there is an on-line geographic information repository containing data about the geographic area being flown over. This includes road data, elevation data, data about physical structures, etc.

The approximation trees described in the introduction are stored in part as relational database content and in part as additional specialized data structures. In this case though, both the nature of the relational tables, queries to the database, and the associated inference mechanisms will all have to be generalized. One reason for this is due to the fact that all relations are assumed to be approximate, therefore the upper and lower approximations have to be represented. This particular type of generalized deductive database is called a *rough relational database* and will be considered in section 6. Rough relational databases, and the semantic and computational mechanisms associated with them, provide us with an efficient means for implementing query/answering systems for approximation trees. How this is done, will be the main topic of the paper.

2 Chapter Outline

In the remaining part of the chapter, we will provide the details for the knowledge representation framework discussed in the introductory section. In section 3, we begin with a brief introduction of some of the basics of rough set theory, primarily to keep the article self-contained for those not familiar with these concepts and techniques. In section 4, we describe a logical language for referring to constituents of rough sets. The language is used as a bridge between more traditional rough set techniques and nomenclature and their use or reference in a logical language. We also define a subset of first-order logic which permits efficient computation of approximation transducers. In section 5, we provide a more detailed description of approximation transducers and include an introductory example. In section 6, we introduce a generalization of deductive databases which permits the use and storage of approximate or rough relations. Section 7 contains the core formal results which provide a semantics for approximation transducers and justifies the computational mechanisms used. The complexity of the approach is also considered. In section 8, we provide a more detailed example of the framework and techniques using traffic congestion as a concept to be modeled. In section 9, we propose an interesting measure of the approximation quality of theories which can be used to compare approximate theories. In section 10, we summarize the results of the chapter and provide a pointer to additional related work described in this volume.

3 Rough Set Theory

In the introductory section we described a framework for self-adaptive and grounded knowledge structures in terms of approximation transducers and trees. One basic premise of the approach was the assumption that approximate primitive concepts could be generated via the application of learning techniques. One particular approach to inducing approximations of concepts is through the use of rough set supervised learning techniques in which sample data is stored in tables and approximate concepts are learned. The result is a concept defined in terms of both a lower and an upper approximation. In this section we will provide a short introduction to a small part of rough set theory and introduce terminology used in the remaining parts of the chapter. We will only briefly mention rough set learning techniques by describing *decision* *systems* which provide the basic structures for rough set learning techniques. Before providing formal definitions, we will first consider an intuitive example from a UAV traffic scenario application.

Example 1. Consider a UAV equipped with a sensor platform which includes a digital camera. Suppose that the UAV task is to recognize various situations on roads. It is assumed that the camera has a particular resolution. It follows that the precise shape of the road cannot be recognized if essential features of the road shape require a higher resolution then that provided by the camera. Figure 2 depicts a view from the UAV's camera, where a fragment of a road is shown together with three cars c1, c2, and c3.

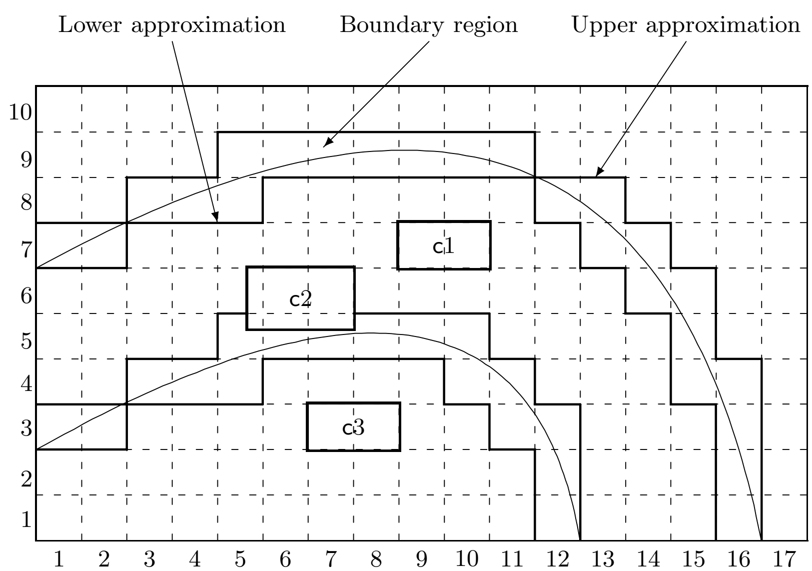

Fig. 2. Sensing a road considered in Example 1

Observe that due to the camera resolution there are collections of points that should be interpreted as being *indiscernible* from each other. The collections of indiscernible points are called *elementary sets*, using rough set terminology. In Figure 2, elementary sets are illustrated by dashed squares and correspond to pixels. Any point in a pixel is not discernible from any other point in the pixel from the perspective of the UAV. Elementary sets are then used to approximate objects that cannot be precisely represented by means of (unions of) elementary sets. For instance, in Figure 2, it can be observed that for some elementary sets one part falls within and the other outside the actual road boundaries (represented by curved lines) simultaneously.

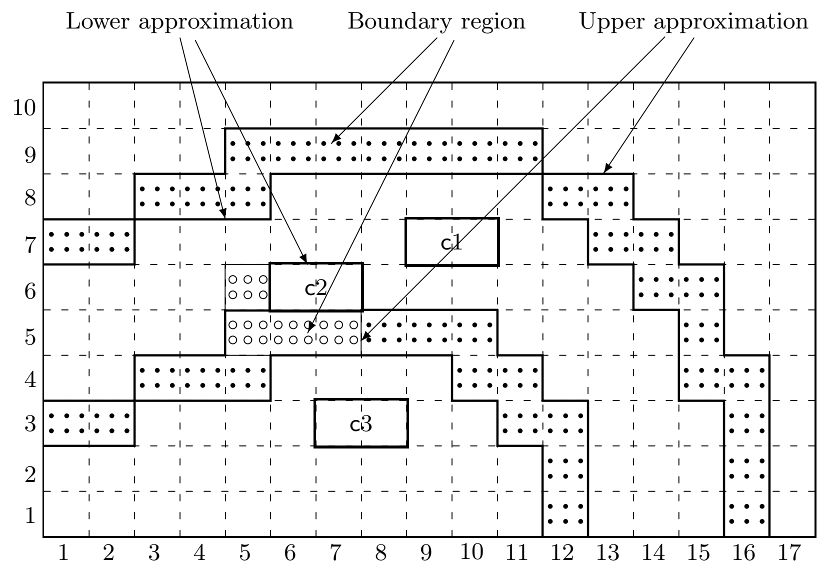

Fig. 3. The approximate view of the road considered in Example 1

Instead of a precise characterization of the road and cars, using rough set techniques, one can obtain approximate characterizations as depicted in Figure 3. Observe that the road sequence is characterized only in terms of a lower and upper approximation of the actual road. A boundary region, containing points that are unknown to be inside or outside of the road's boundaries, is characterized by a collection of elementary sets marked with dots inside. Cars c1 and c3 are represented precisely, while car c2 is represented by its lower approximation (the thick box denoted by c2) and by its upper approximation (the lower approximation together with the region containing elementary sets marked by hollow dots inside). The region of elementary sets marked by hollow dots inside represents the boundary region of the car.

The lower approximation of a concept represents points that are known to be part of the concept, the boundary region represents points that might or might not be part of the concept, and the complement of the upper approximation represents points that are known not to be part of the concept. Consequently, car c1 is characterized as being completely on the road (inside the road's boundaries); it is unknown whether car c2 is completely on the road and car c3 is known to be outside, or off the road.■

As illustrated in Example 1, the rough set philosophy is founded on the assumption that we associate some information (data, knowledge) with every object of the universe of discourse. This information is often formulated in terms of attributes about objects. Objects characterized by the same information are interpreted as indiscernible (similar) in view of the available information about them. An indiscernibility relation, generated in this manner from the attribute/value pairs associated with objects, provides the mathematical basis of rough set theory.

Any set of all indiscernible (similar) objects is called an *elementary set*, and forms a basic granule (atom) of knowledge about the universe. Any union of some elementary sets in a universe is referred to as a *crisp (precise) set*; otherwise the set is referred to as being a *rough (imprecise, vague) set*. In the latter case, two separate unions of elementary sets can be used to approximate the imprecise set, as we have seen in the example above.

Consequently, each rough set has what are called boundary-line cases, i.e., objects which cannot with certainty be classified either as members of the set or of its complement. Obviously, crisp sets have no boundary-line elements at all. This means that boundary-line cases cannot be properly classified by employing only the available information about objects.

The assumption that objects can be *observed* only through the information available about them leads to the view that knowledge about objects has granular structure. Due to this granularity, some objects of interest cannot always be discerned given the information available, therefore the objects appear as the same (or similar). As a consequence, vague or imprecise concepts, in contrast to precise concepts, cannot be characterized solely in terms of information about their elements since elements are not always discernible from each other. In the proposed approach, we assume that any vague or imprecise concept is replaced by a pair of precise concepts called the lower and the upper approximation of the vague or imprecise concept. The lower approximation consists of all objects which with certainty belong to the concept and the upper approximation consists of all objects which have a possibility of belonging to the concept.

The difference between the upper and the lower approximation constitutes the boundary region of a vague or imprecise concept. Additional information about attribute values of objects classified as being in the boundary region of a concept may result in such objects being re-classified as members of the lower approximation or as not being included in the concept. Upper and lower approximations are two of the basic operations in rough set theory.

3.1 Information Systems and Indiscernibility

One of the basic concepts of rough set theory is the indiscernibility relation which is generated using information about particular objects of interest. Information about objects is represented in the form of a set of attributes and their associated values for each object. The indiscernibility relation is intended to express the fact that, due to lack of knowledge, we are unable to discern some objects from others simply by employing the available information about those objects. In general, this means that instead of dealing with each individual object we often have to consider clusters of indiscernible objects as fundamental concepts of our theories.

Let us now present this intuitive picture about rough set theory more formally.

Definition 1. An *information system* is any pair $\mathcal{A} = \langle U, A \rangle$ where U is a non-empty finite set of *objects* called the *universe* and A is a non-empty finite set of *attributes* such that $a : U \to V_a$ for every $a \in A$. The set V_a is called the *value set* of a. By $Inf_B(x) = \{\langle a, a(x) \rangle : a \in B\}$, we denote the *information signature of x with respect to B*, where $B \subseteq A$ and $x \in U$.■

Note that in this definition, attributes are treated as functions on objects, where $a(x)$ denotes the value the object x has for the attribute a.

Any subset B of A determines a binary relation $IND_{\mathcal{A}}(B) \subseteq U \times U$, called an indiscernibility relation, defined as follows.

Definition 2. Let $\mathcal{A} = \langle U, A \rangle$ be an information system and let $B \subseteq A$. By the *indiscernibility relation determined by B*, denoted by $IND_{\mathcal{A}}(B)$, we understand the relation

$$IND_{\mathcal{A}}(B) = \{(x, x') \in U \times U : \forall a \in B.[a(x) = a(x')]\}.$$

If $(x, y) \in IND_{\mathcal{A}}(B)$ we say that x and y are *B-indiscernible*. Equivalence classes of the relation $IND_{\mathcal{A}}(B)$ (or blocks of the partition U/B) are referred to as *B-elementary sets*. The unions of B-elementary sets are called *B-definable sets*.■

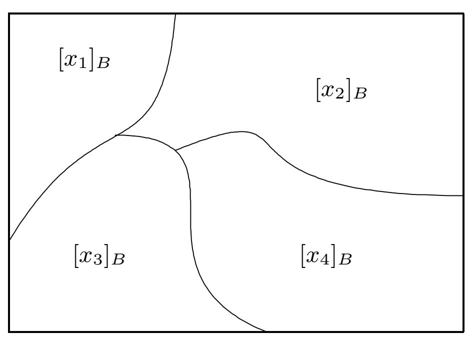

Fig. 4. A rough partition $IND_{\mathcal{A}}(B)$.

Observe that $IND_{\mathcal{A}}(B)$ is an equivalence relation. Its classes are denoted by $[x]_B$. By U/B we denote the partition of U defined by the indiscernibility relation $IND_{\mathcal{A}}(B)$. For example, in Figure 4, the partition of U defined by an indiscernibility relation $IND_{\mathcal{A}}(B)$ contains four equivalence classes, $[x_1]_B, [x_2]_B, [x_3]_B$ and $[x_4]_B$. An example of a B-definable set would be $[x_1]_B \cup [x_4]_B$, where $[x_1]_B$ and $[x_4]_B$ are B-elementary sets.

In Example 1, the indiscernibility relation is defined by a partition corresponding to pixels represented in Figures 2 and 3 by squares with dashed borders. Each square represents an elementary set. In the rough set approach the elementary sets are the basic building blocks (concepts) of our knowledge about reality.

The ability to discern between perceived objects is also important for constructing many entities like reducts, decision rules, or decision algorithms which are used in rough set based learning techniques. In the classical rough set approach the *discernibility relation*, $DIS_{\mathcal{A}}(B)$, is defined as follows.

Definition 3. Let $\mathcal{A} = \langle U, A \rangle$ be an information system and $B \subseteq A$. The *discernibility relation* $DIS_{\mathcal{A}}(B) \subseteq U \times U$ is defined as $(x, y) \in DIS_{\mathcal{A}}(B)$ if and only if $(x, y) \notin IND_{\mathcal{A}}(B)$. ■

3.2 Approximations and Rough Sets

Let us now define approximations of sets in the context of information systems.

Definition 4. Let $\mathcal{A} = \langle U, A \rangle$ be an information system, $B \subseteq A$ and $X \subseteq U$. The *B-lower approximation* and *B-upper approximation* of X, denoted by X_{B^+} and $X_{B^\oplus}$ respectively, are defined by $X_{B^+} = \{x : [x]_B \subseteq X\}$ and $X_{B^\oplus} = \{x : [x]_B \cap X \neq \emptyset\}$.■

The B-lower approximation of X is the set of all objects which can be classified with certainty as belonging to X just using the attributes in B to discern distinctions.

Definition 5. The set consisting of objects in the *B-lower approximation* X_{B^+} is also called the *B-positive region of* X. The set $X_{B^-} = U - X_{B^\oplus}$ is called the *B-negative region of* X. The set $X_{B^\pm} = X_{B^\oplus} - X_{B^+}$ is called the *B-boundary region of* X.■

Observe that the positive region of X consists of objects that can be classified with certainty as belonging to X using attributes from B. The negative region of X consists of those objects which can be classified with certainty as not belonging to X using attributes from B. The B-boundary region of X consists of those objects that cannot be classified unambiguously as belonging to X using attributes from B.

For example, in Figure 5, The B-lower approximation of the set X, X_{B^+}, is $[x_2]_B \cup [x_4]_B$. The B-upper approximation, $X_{B^\oplus}$, is $[x_1]_B \cup [x_2]_B \cup [x_4]_B \equiv [x_1]_B \cup X_{B^+}$. The B-boundary region, $X_{B^\pm}$, is $[x_1]_B$. The B-negative region of X, X_{B^-}, is $[x_3]_B \equiv U - X_{B^\oplus}$.

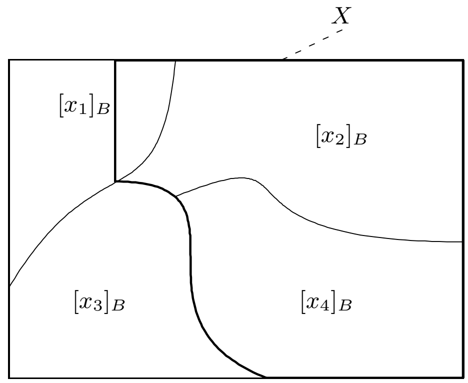

Fig. 5. A rough partition $IND_{\mathcal{A}}(B)$ and an imprecise set X.

3.3 Decision Systems and Supervised Learning

Rough set techniques are often used as a basis for supervised learning using tables of data. In many cases the target of a classification task, that is, the family of concepts to be approximated, is represented by an additional attribute called a decision attribute. Information systems of this kind are called decision systems.

Definition 6. Let $\langle U, A\rangle$ be an information system. A *decision system* is any system of the form $\mathcal{A} = \langle U, A, d\rangle$, where $d \notin A$ is the *decision attribute* and A is a set of *conditional attributes*, or simply *conditions*.■

Let $\mathcal{A} = \langle U, A, d\rangle$ be given and let $V_d = \{v_1, \ldots, v_{r(d)}\}$. Decision d determines a partition $\{X_1, \ldots, X_{r(d)}\}$ of the universe U, where $X_k = \{x \in U : d(x) = v_k\}$ for $1 \leq k \leq r(d)$. The set X_i is called the i-th *decision class* of $\mathcal{A}$. By $X_{d(u)}$ we denote the decision class $\{x \in U : d(x) = d(u)\}$, for any $u \in U$.

One can generalize the above definition to the case of decision systems of the form $\mathcal{A} = \langle U, A, D\rangle$ where the set $D = \{d_1, ...d_k\}$ of decision attributes and A are assumed to be disjoint. Formally this system can be treated as the decision system $\mathcal{A} = \langle U, C, d_D\rangle$ where $d_D(x) = (d_1(x), ..., d_k(x))$ for $x \in U$.

A decision table can be identified as a representation of raw data (or training samples in machine learning) which is used to induce concept approximations in a process known as supervised learning. Decision tables themselves are defined in terms of decision systems. Each row in a decision table represents one training sample. Each column in the table represents a particular attribute in A, with the exception of the first column which represents objects in U and selected columns representing the decision attribute(s).

There is a wide variety of techniques that have been developed for inducing approximations of concepts relative to various subsets of attributes in decision systems. The methods are primarily based on viewing tables as a type of boolean formula, generating reducts for these formulas, which are concise descriptions of tables with redundancies removed, and generating decision rules from these formula descriptions. The decision rules can be used as classifiers or as representations of lower and upper approximations of the induced concepts. In this chapter, we will not pursue these techniques.

What is important for understanding our framework is the fact that these techniques exist, they are competitive with other learning techniques and often more efficient, and, given raw sample data, such as low level feature data from an image processing system represented as tables, primitive concepts can be induced or learned. These concepts are characterized in terms of upper and lower approximations and represent grounded contextual approximations of concepts and relations from the application domain. This is all we need to assume to construct grounded approximation transducers and to recursively construct approximation trees.

4 A Logical Language for Rough Set Concepts

One final component which bridges the gap between more conventional rough set techniques and logical languages used to specify and compute with approximation transducers is a logical vocabulary for referring to constituent components of a rough set when viewed as a relation or property in a logical language. Note that this particular ontological policy provides the right syntactical characterization of rough set concepts we require for our framework. One could also envision a different ontological policy, with a higher level of granularity for instance, that could be used for other purposes.

In order to construct a logical language for referring to constituent components of rough concepts, we introduce the following relation symbols for any rough relation R (see Figure 6):

- R^+ – represents the positive facts known about the relation. R^+ corresponds to the lower approximation of R. R^+ is called the *positive region (part)* of R.
- R^- – represents the negative facts known about the relation. R^- corresponds to the complement of the upper approximation of R. R^- is called the *negative region (part)* of R.
- $R^\pm$ – represents the unknown facts about the relation. $R^\pm$ corresponds to the set difference between the upper and lower approximations to R. $R^\pm$ is called the *boundary region (part)* of R.
- $R^\oplus$ – represents the positive facts known about the relation together with the unknown facts. $R^\oplus$ corresponds to the upper approximation to R. $R^\oplus$ is called the *positive-boundary region (part)* of R.
- $R^\ominus$ – represents the negative facts known about the relation together with the unknown facts. $R^\ominus$ corresponds to the upper approximation of

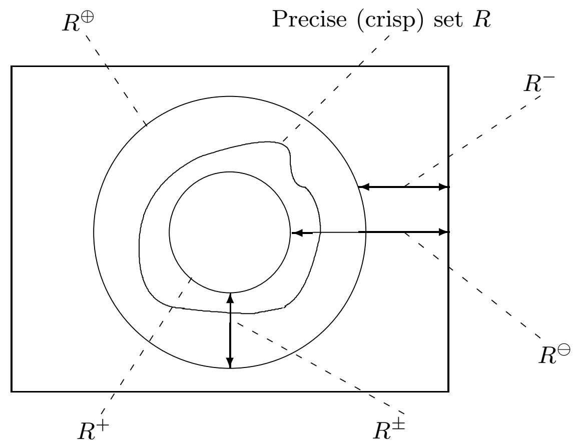

Fig. 6. Representation of a rough set in logic

the complement of R. $R^{\ominus}$ is called the *negative-boundary region (part)* of R.

For the sake of simplicity, in the rest of this chapter we will assume that a theory defines only one intensional rough relation. We shall use the notation $Th(R; R_1, \ldots, R_n)$ to indicate that R is approximated by Th, where the $R_1, \ldots, R_n$ are the input concepts to a transducer. We also assume that negation occurs only directly before relation symbols.[2]

We write $Th^{+}(R; R_1, \ldots, R_n)$ (or Th^{+}, for short) to denote theory Th with all positive literals R_i substituted by ${R_i}^{+}$ and all negative literals substituted by R_i^{-}. Similarly we write $Th^{\oplus}(R; R_1, \ldots, R_n)$ (or $Th^{\oplus}$, in short) to denote theory Th with all positive literals R_i substituted by $R_i^{\oplus}$ and all negative literals substituted by ${R_i}^{\ominus}$. We often simplify the notation using the equivalences $\neg R^{-}(\bar{x}) \equiv R^{\oplus}(\bar{x})$ and $\neg R^{+}(\bar{x}) \equiv R^{\ominus}(\bar{x})$.

4.1 Additional Notation and Preliminaries

In order to guarantee that both the inference mechanism specified for querying approximation trees and the process used to compute approximate relations using approximation transducers are efficient, we will have to place a number of syntactic constraints on the local theories used in approximation transducers. The following definitions will be useful for that purpose.

[2] Any first- or second-order formula can be equivalently transformed into this form.

Definition 7. A predicate variable R occurs *positively* (resp. *negatively*) in a formula Φ if the prenex and conjunctive normal form[3] of Φ contains a literal of the form $R(\bar{t})$ (resp. $\neg R(\bar{t})$). A formula Φ is said to be *positive* (resp. *negative*) w.r.t. R iff all occurrences of R in Φ are positive (resp. negative).■

Definition 8. A formula is called a *semi-Horn rule* (or *rule*, for short) w.r.t. relation symbol R provided that it is in one of the following forms:

$$\forall \bar{x}.[R(\bar{x}) \rightarrow \Psi(R, R_1, \ldots R_n)] \tag{1}$$

$$\forall \bar{x}.[\Psi(R, R_1, \ldots R_n) \rightarrow R(\bar{x})], \tag{2}$$

where Ψ is an arbitrary classical first-order formula positive w.r.t. R and $\bar{x}$ is an arbitrary vector of variable symbols. If formula Ψ of a rule does not contain R, the rule is called *non-recursive w.r.t. R*.■

Example 2. The first of the following formulas is a (recursive) semi-Horn rule w.r.t. R, while the second one is a non-recursive semi-Horn rule w.r.t. R:

$$\forall x, y.[\exists u.(R(u, y) \vee \exists z.(S(z, x, z) \wedge \forall t.R(z, t)))] \rightarrow R(x, y)$$
$$\forall x, y.[\exists u.(T(u, y) \vee \exists z.(S(z, x, z) \wedge \forall t.Q(z, t)))] \rightarrow R(x, y)$$

The following formula is not a semi-Horn rule, since R appears negatively in the lefthand side of the rule.

$$\forall x, y.[\exists u.(\neg R(u, y) \vee \exists z.(S(z, x, z) \wedge \forall t.R(z, t)))] \rightarrow R(x, y)$$

■

Observe that one could also deal with dual forms of the rules (1) and (2), obtained by replacing relation R by $\neg R$. It is sometimes more convenient to use rules of such a form. For instance, one often uses rules like "if an object on a highway is a car and is *not* abnormal then it moves". Of course, the results we present can easily be adapted to such a situation.

We often write rules of the form (1) and (2) without initial universal quantifiers, understanding that the rules are always implicitly universally quantified.

5 Approximation Transducers

As stated in the introduction, an approximation transducer provides a means of generating or defining an approximate relation (the output) in terms of other approximate relations (the input) using various dependencies between the input and the output.[4] The set of dependencies is in fact a logical theory

[3] i.e. the form with all quantifiers in the prefix of the formula and the quantifier free part of the formula in the form of a conjunction of clauses.

[4] The technique also works for one or more approximate relations being generated as output, but for clarity of presentation, we will describe the techniques using a single output relation.

where each dependency is represented as a logical formula in a first-order logical language. Syntactic restrictions can be placed on the logical theory to insure efficient generation of output.

Since we are dealing with approximate relations, both the inputs and output are defined in terms of upper and lower approximations. In section 4, we introduced a logical language for referring to different constituents of a rough relation. It is not necessary to restrict the logical theory to just the relations specified in the input and output for a particular transducer. Other relations may be used since they are assumed to be defined or definitions can be generated simultaneously with the generation of the particular output in question. In other words, it is possible to define an approximation network rather than a tree, but for this presentation, we will stick to the tree-based approach. The network approach is particularly interesting because it allows for limited forms of feedback across abstraction levels in the network.

The main idea is depicted in Fig. 7. Suppose one would like to define an approximation of a relation R in terms of a number of other approximate relations $R_1, \ldots, R_k$. It is assumed that $R_1, \ldots, R_k$ consist of either primitive relations acquired via a learning phase or approximate relations that have been generated recursively via other transducers or combinations of transducers.

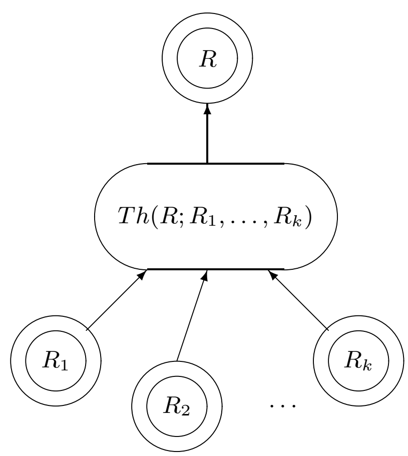

Fig. 7. Transformation of rough relations by first-order theories

The local theory $Th(R; R_1, \ldots, R_k)$ is assumed to contain logical formulas relating the input to the output and can be acquired through a knowledge acquisition process with domain experts or even through the use of inductive logic programming techniques. Generally the formulas in the logical theory are provided in the form of rules representing some sufficient and necessary conditions for the output relation in addition to possibly other conditions. The local theory should be viewed as a logical template describing a dependency structure between relations.

The actual transduction process which generates the approximate definition of relation R uses the logical template and contextualizes it with the actual contextual approximate relations provided as input. The result of the transduction process provides a definition of both the upper and lower approximation of R as follows,

- The lower approximation is defined as the least model for R w.r.t. the theory

$$Th^{+}(R; R_1, \ldots, R_k),$$

- and the upper approximation is defined as the greatest model for R w.r.t. the theory

$$Th^{\oplus}(R; R_1, \ldots, R_k),$$

where Th^{+} and $Th^{\oplus}$ denote theories obtained from Th by replacing crisp relations by their corresponding approximations (see section 4). As a result one obtains an approximation of R defined as a rough relation. Note that appropriate syntactic restrictions are placed on the theory so coherence conditions can be generated which guarantee the existence of the least and the greatest model of the theory and its consistency with the approximation tree in which its transducer is embedded. For details, see section 7.

Implicit in the approach is a notion of abstraction hierarchies where one can recursively define more abstract approximate relations in terms of less abstract approximations by combining different transducers. The result is one or more approximation trees. This intuition has some similarity with the idea of layered learning (see, e.g., [12]). The technique also provides a great deal of locality and modularity in representation although it does not force this on the user since networks violating locality can be constructed. If one starts to view an approximation transducer or sub-tree of approximation transducers as simple or complex agents responsible for the management of particular relations and their dependencies, this then has some correspondence with the methodology strongly advocated, e.g., in [7].

The ability to continually apply learning techniques to the primitive relations in the network and to continually modify the logical theories which are constituent parts of transducers provides a great deal of elaboration tolerance and elasticity in the knowledge representation structures. If the elaboration is automated for an intelligent artifact using these structures, the claim can be made that these knowledge structures are self-adaptive, although a great deal more work would have to be done to realize this practically.

5.1 An Introductory Example

In this section, we provide an example for a single approximation transducer describing some simple relationships between objects on a road. Assume we are provided with the following rough relations:

- $V(x, y)$ – there is a visible connection between objects x and y.
- $S(x, y)$ – the distance between objects x and y is small.
- $E(x, y)$ – objects x and y have equal speed.

We can assume that these relations were acquired using a supervised learning technique where sample data was generated from video logs provided by the UAV when flying over a particular road system populated with traffic, or that the relations were defined as part of an approximation tree using other approximation transducers.

Suppose we would like to define a new relation C denoting that its arguments, two objects on the road, are connected. It is assumed that we as knowledge engineers or domain experts have some knowledge of this concept. Consider, for example, the following local theory $Th(C; V, S, E)$ approximating C:

$$\forall x, y.[V(x, y) \rightarrow C(x, y)] \tag{3}$$

$$\forall x, y.[C(x, y) \rightarrow (S(x, y) \wedge E(x, y))]. \tag{4}$$

The former provides a sufficient condition for C and the latter a necessary condition. Imprecision in the definition is caused by the following facts:

- the input relations V, S and E are imprecise (rough, non-crisp)
- the theory $Th(C; V, S, E)$ does not describe relation C precisely, as there are many possible models for C.

We then accept the least model for C w.r.t. theory $Th(C; V^+, S^+, E^+)$ as the lower approximation of C and the upper approximation as the greatest model for C w.r.t. theory $Th(C; V^\oplus, S^\oplus, E^\oplus)$.

It can now easily be observed (and, in fact, be computed efficiently), that one can obtain the following definitions of the lower and upper approximations of C:

$$\forall x, y.[C^+(x, y) \equiv V^+(x, y)] \tag{5}$$

$$\forall x, y.[C^\oplus(x, y) \equiv (S^\oplus(x, y) \wedge E^\oplus(x, y))]. \tag{6}$$

Relation C can then be used, e.g., while querying the rough knowledge database containing this approximation tree or for defining new approximate concepts, provided that it is coherent with the database contents. In this case, the coherence conditions, which guarantee the consistency of the generated relation with the rest of the database (approximation tree), are expressed by the following formulas:

$$\forall x, y.[V^{+}(x, y) \rightarrow (S^{+}(x, y) \wedge E^{+}(x, y))]$$

$$\forall x, y.[V^{\oplus}(x, y) \rightarrow (S^{\oplus}(x, y) \wedge E^{\oplus}(x, y))].$$

The coherence conditions can also be generated in an efficient manner provided certain syntactic constraints are applied to the local theories in an approximation transducer.

6 Rough Relational Databases

In order to compute the output of an approximation transducer, syntactic characterizations of both the upper and lower approximations of the output relation relative to the substituted local theories are generated. Depending on the expressiveness of the local theory used in a transducer, the results are either first-order formulas or fixpoint formulas. These formulas can then be used to query a rough relational database in an efficient manner using a generalization of results from [4] and traditional relational database theory.

In this section, we define what rough relational databases are and consider their use in the context of approximation transducers and trees. In the following section we provide the semantic and computational mechanisms used to generate output relations of approximation transducers, check the coherence of the output relations, and ask queries about any approximate relation in an approximation tree.

Definition 9. A *rough relational database* B, is a first order structure $\langle U, r_1^{a_1}, \ldots, r_k^{a_k}, c_1, \ldots, c_l \rangle$, where

- U is a finite set,
- for $1 \leq i \leq k$, $r_i^{a_i}$ is an a_i-argument rough relation on U, i.e. $r_i^{a_i}$ is given by its lower approximation $r_i^{a_i +}$, and upper approximation $r_i^{a_i \oplus}$
- $c_1, \ldots, c_l \in U$ are constants.

By a *signature* of B we mean a signature containing relation symbols $R_1^{a_1}, \ldots,$ $R_k^{a_k}$ and constant symbols $c_1, \ldots, c_l$ together with equality $=$.■

According to the terminology accepted in the literature, a deductive database consists of two parts: an extensional and intensional database. The extensional database is usually equivalent to a traditional relational database and the intensional database contains a set of definitions of relations in terms of rules that are not explicitly stored in the database. In what follows, we shall also use the terminology *extensional* (*intensional*) rough relation to indicate that the relation is stored or defined, respectively.

Note that intensional rough relations are defined in this framework by means of theories that do not directly provide us with explicit definitions of the relations. These are the local theories in approximation transducers. In fact we apply the methodology developed in [4] which is based on the use of quantifier elimination applied to logical queries to conventional relational databases. The work in [4] is generalized here to rough relational databases.

According to [4], the computation process can be described in two stages. In the first stage, we provide a PTime compilation process which computes explicit definitions of intensional rough relations. In our case, we would like to compute the explicit definitions of the upper and lower approximations of a relation output from an approximation transducer. In the second stage, we use the explicit definitions of the upper and lower approximations of the intensional relation generated in the first stage to compute suitable relations in the rough relational database that satisfy the local theories defining the relations.

We also have to check whether such relations exist relative to the rough relational database in question. This is done by checking so-called coherence conditions. It may be the case that the complex query to the rough relational database which includes the constraints in the local theory associated with a transducer is not consistent with relations already defined in the database itself. Assuming the query is consistent, we know that the output relation for the approximation transducer in question exists and can now compute the answer.

Both checking that the theory is coherent and computing the output relation can be done efficiently because these tasks reduce to calculating fixpoint queries to relational databases over finite domains, a computation which is in PTime (see, e.g., [5]). Observe that the notion of coherence conditions is adapted in this paper to deal with rough relations rather than with precise relations as done in [4].

7 Approximation Transducer Semantics and Computation Mechanisms

Our specific target is to define a new relation, say R, in terms of some additional relations $R_1, \ldots, R_n$ and a local logical theory $Th(R; R_1, \ldots, R_n)$ representing knowledge about R and its relation to $R_1, \ldots, R_n$. The output of the transduction process results in a definition of R^+, the lower approximation of R, as the least model of $Th^+(R; R_1, \ldots, R_n)$ and $R^\oplus$, the upper approximation of R, as the greatest model of $Th^\oplus(R; R_1, \ldots, R_n)$. The following problems must be addressed:

- Is $Th(R; R_1, \ldots, R_n)$ consistent with the database?
- Does a least and greatest relation R^+ and $R^\oplus$ exist, which satisfies $Th^+(R; R_1, \ldots R_n)$ and $Th^\oplus(R; R_1, \ldots R_n)$, respectively?
- Is the complexity of the mechanisms used to answer the above questions and to calculate suitable approximations R^+ and $R^\oplus$ reasonable from a pragmatic perspective?

In general, consistency is not guaranteed. Moreover, the above problems are generally NPTime-complete (over finite models). However, quite similar questions were addressed in [4] and a rich class of formulas has been isolated for which the consistency problem and the other problems can be resolved in PTime. In what follows, we will use results from [4] and show that a subset of semi-Horn formulas from [4], which we call *semi-Horn rules* (or just rules, for short) and which are described in section 4.1, guarantees the following:

- The coherence conditions for $Th(R; R_1, \ldots, R_n)$ can be computed and checked in polynomial time;
- The least and the greatest relations R^+ and $R^\oplus$, satisfying $Th^+(R; R_1, \ldots, R_n)$ and $Th^\oplus(R; R_1, \ldots, R_n)$, respectively, always exist provided that the coherence conditions are satisfied;
- The time and space complexity of calculating suitable approximations R^+ and $R^\oplus$ is polynomial w.r.t. the size of the database and that of calculating their symbolic definitions is polynomial in the size of $Th(R; R_1, \ldots, R_n)$.

In view of these positive results, we will restrict the set of formulas used in local theories in transducers to (finite) conjunctions of semi-Horn rules as defined in section 4.1. All theories considered in the rest of the paper are assumed to be semi-Horn in the sense of Definition 8.

The following lemmas (Lemma 1 and 2) provide us with a formal justification of Definition 10 which follows. Let us first deal with non-recursive rules[5].

Lemma 1 is based on a lemma of Ackermann (see, e.g., [3]) and results of [4].

Lemma 1. Assume that $Th(R; R_1, \ldots, R_n)$ consists of the following rules:

$$\forall \bar{x}.[R(\bar{x}) \rightarrow \Phi_i(R_1, \ldots, R_n)], \tag{7}$$

$$\forall \bar{x}.[\Psi_j(R_1, \ldots, R_n) \rightarrow R(\bar{x})], \tag{8}$$

for $i \in I$, $j \in J$, where I, J are finite, nonempty sets and for all $i \in I$ and $j \in J$, formulas Φ_i and Ψ_j do not contain occurrences of R. Then there exist the least and the greatest R satisfying (7) and (8). The least such R is defined by the formula:

$$R(\bar{x}) \equiv \bigvee_{j \in J} \Psi_j(R_1, \ldots, R_n) \tag{9}$$

and the greatest such R is defined by the formula:

$$R(\bar{x}) \equiv \bigwedge_{i \in I} \Phi_i(R_1, \ldots, R_n), \tag{10}$$

[5] In fact, Lemma 1 follows easily from Lemma 2 by observing that fixpoint formulas (14), (15) and (16) reduce in this case to first-order formulas (9), (10) and (11), respectively. However, reductions to classical first-order formulas are worth a separate treatment as these are less complex and easier to deal with.

provided that the following coherence condition is satisfied in the database:

$$\forall \bar{x}. \left[\bigvee_{j \in J} \Psi_j(R_1, \ldots, R_n) \rightarrow \bigwedge_{i \in I} \Phi_i(R_1, \ldots, R_n) \right] \tag{11}$$

Proof. Follows easily, e.g., from Theorem 5.3 of [4].■

Denote by $\mu S.\alpha(S)$ the least and by $\nu S.\alpha(S)$ the greatest simultaneous fixpoint operator of $\alpha(S)$ (for the definition of fixpoints see, e.g., [5]). Then in the case of recursive theories we can prove the following lemma, based on the fixpoint theorem of [8] and results of [4].

Lemma 2. Assume that $Th(R; R_1, \ldots, R_n)$ consists of the following rules:

$$\forall \bar{x}.[R(\bar{x}) \rightarrow \Phi_i(R, R_1, \ldots, R_n)], \tag{12}$$

$$\forall \bar{x}.[\Psi_j(R, R_1, \ldots, R_n) \rightarrow R(\bar{x})], \tag{13}$$

for $i \in I$, $j \in J$, where I, J are finite, nonempty sets. Then there exist the least and the greatest R satisfying formulas (12) and (13). The least such R is defined by the formula:

$$R(\bar{x}) \equiv \mu R(\bar{x}).[\bigvee_{j \in J} \Psi_j(R, R_1, \ldots, R_n)] \tag{14}$$

and the greatest such R is defined by the formula:

$$R(\bar{x}) \equiv \nu R(\bar{x}).[\bigwedge_{i \in I} \Phi_i(R, R_1, \ldots, R_n)] \tag{15}$$

provided that the following coherence condition holds:

$$\forall \bar{x}. \left[\mu R(\bar{x}).[\bigvee_{j \in J} \Psi_j(R, R_1, \ldots, R_n)] \rightarrow \nu R(\bar{x}).[\bigwedge_{i \in I} \Phi_i(R, R_1, \ldots, R_n)] \right] \tag{16}$$

Proof. Follows easily, e.g., from Theorem 5.2 of [4].■

The following definition provides us with a semantics of semi-Horn rules used as local theories in rough set transducers.

Definition 10. Let B be a rough relational database with extensional relation symbols $R_1, \ldots, R_n$ and let R be an intensional relation symbol.

By an *approximation transducer* we intend the input to be $R_1, \ldots, R_n$, the output to be R and the local transducer theory to be a first-order theory $Th(R; R_1, \ldots, R_n)$ expressed by rules of the form (12)/ (13) or (7)/(8). Under these restrictions,

- the lower approximation of R is defined as the least relation R satisfying $Th(R; R_1, \ldots, R_n)$, i.e. the relation defined by formula $(9)^+$ or $(14)^+$, respectively, with $R_1, \ldots, R_n$ substituted as described in section 4

- the upper approximation of R is defined as the greatest relation R satisfying $Th(R; R_1, \dots, R_n)$ i.e. the relation defined by formula $(10)^{\oplus}$ or $(15)^{\oplus}$, respectively, with $R_1, \dots, R_n$ substituted as described in section 4,

provided that the respective coherence conditions $(11)^{+}$ or $(16)^{+}$, for the lower approximation, and $(11)^{\oplus}$ or $(16)^{\oplus}$, for the upper approximation, are satisfied in database B.■

Observe that we place a number of restrictions on this definition that can be relaxed, such as restricting use of relation symbols in the local theory of the transducer to be crisp. This excludes use of references to constituent components of other rough relations. In addition, since the output relation of a transducer can be represented explicitly in the rough relational database, approximation trees consisting of combinations of transducers are well defined.

7.1 The Complexity of the Approach

This framework is presented in the context of relational databases with finite domains with some principled generalizations. In addition, both explicit definitions of approximations to relations and associated coherence conditions are expressed in terms of classical first-order or fixpoint formulas. Consequently, computing the approximations and checking coherence conditions can be done in time polynomial in the size of the database (see, e.g., [5]).

In addition, the size of explicit definitions of approximations and coherence conditions is linear in the size of the local theories defining the approximations. Consequently, the proposed framework is acceptable from the point of view of a formal complexity analysis. This serves as a useful starting point for efficient implementation of the techniques. It is clear though, that for very large databases of this type, additional optimization methods would be desirable.

8 A Congestion Example

In this section, we provide an example from the UAV-traffic domain which demonstrates one approach to the problem of defining the concept of traffic congestion using the proposed framework. We begin by assuming the following relations and constants exist:

- l – is a traffic lane on the road.
- $inFOA(l)$ – denotes whether lane l is in the focus of the UAV camera.
- $inROI(x)$ – denotes whether a vehicle x is in a region of interest.
- $Speed(x, z)$ – denotes the approximate speed of x, where $z \in \{\mathsf{low}, \mathsf{medium}, \mathsf{high}, \mathsf{unknown}\}$.
- $Distance(x, y, z)$ – denotes the approximate distance between vehicles x and y, where $z \in \{\mathsf{small}, \mathsf{medium}, \mathsf{large}, \mathsf{unknown}\}$.

- $Between(z, x, y)$ – denotes whether vehicle z is between vehicles x and y.
- $Number(x, y, z)$ – denotes the approximate number of vehicles between vehicles x and y occurring in the region of interest, where $z \in \{\mathsf{small}, \mathsf{medium}, \mathsf{large}, \mathsf{unknown}\}$,
- $TrafficCong(l)$ – denotes whether there is traffic congestion in lane l.

We define traffic congestion by the following formula:

$$\begin{aligned} TrafficCong(l) \equiv\ & inFOA(l) \land \\ & \exists x, y.[inROI(x) \land inROI(y) \land Number(x, y, \mathsf{large}) \land \\ & \forall z.(Between(z, x, y) \to Speed(z, \mathsf{low})) \land \\ & \forall z.(Between(z, x, y) \to \exists t.(Distance(z, t, \mathsf{small})))]. \end{aligned} \tag{17}$$

Observe that formula (17) contains concepts that are not defined precisely. However, for the example, we assume that the underlying database contains approximations of these concepts. We can then use the approximated concepts and replace formula (17) with the following two formulas representing the lower and upper approximation of the target concept:

$$\begin{aligned} & TrafficCong^{+}(l) \equiv \\ & \quad \exists x, y.[inROI^{+}(x) \land inROI^{+}(y) \land Number^{+}(x, y, \mathsf{large}) \land \\ & \quad \forall z.(Between^{\oplus}(z, x, y) \to Speed^{+}(z, \mathsf{low})) \land \\ & \quad \forall z.(Between^{\oplus}(z, x, y) \to \exists t.Distance^{+}(z, t, \mathsf{small}))] \end{aligned} \tag{18}$$

$$\begin{aligned} & TrafficCong^{\oplus}(L) \equiv \\ & \quad \exists x, y.[inROI^{\oplus}(x) \land inROI^{\oplus}(y) \land Number^{\oplus}(x, y, \mathsf{large}) \land \\ & \quad \forall z.(Between^{+}(z, x, y) \to Speed^{\oplus}(z, \mathsf{low}) \land \\ & \quad \forall z.(Between^{+}(z, x, y) \to \exists t.Distance^{\oplus}(z, t, \mathsf{small}))] \end{aligned} \tag{19}$$

These formulas can be automatically generated using the techniques described previously.

It can now be observed that formula (17) defines a cluster of situations that can be considered as traffic congestions. Namely, small deviations of data do not have a substantial impact on the target concept. This is a consequence of the fact that in (17) we refer to values that are also approximated such as low, small and large. Thus small deviations of vehicle speed or distance between vehicles usually do not change the qualitative classification of these notions.

Let us denote deviations of data by dev with suitable indices. Now, assuming that the deviations satisfy the following properties:

$$\begin{aligned} & x' \in dev_{inROI}(x) \equiv [inROI^{+}(x) \to inROI^{+}(x')] \\ & x' \in dev_{Speed}(x) \equiv [Speed^{+}(x, \mathsf{low}) \to Speed^{+}(x', \mathsf{low})] \\ & (x', y') \in dev_{Number}(x, y) \equiv \\ & \quad [Number^{+}(x, y, \mathsf{large}) \to Number^{+}(x', y', \mathsf{large})] \end{aligned} \tag{20}$$

$$(x', y') \in dev_{Distance}(x, y) \equiv$$
$$[Distance^{+}(x, y, \mathsf{small}) \rightarrow Distance^{+}(x', y', \mathsf{small})]$$
$$(z', x', y') \in dev_{Between}(z, x, y) \equiv$$
$$[Between^{+}(z, x, y) \rightarrow Between^{+}(z', x', y')],$$

one can conclude that:

$$[\mathit{TrafficCong}^{+}(l) \wedge l' \in dev_{\mathit{TrafficCong}}(l)] \rightarrow \mathit{TrafficCong}^{+}(l'),$$

where $dev_{\mathit{TrafficCong}}(l)$ denotes the set of all situations obtained by deviations of l satisfying conditions expressed by (20).

The above reasoning schema is then robust w.r.t. small deviations of input concepts. In fact, any approximation transducer defined using purely logical means, enjoys this property since small deviations of data, by not changing basic properties, do not change the target concept.

A formal framework which includes the topics of robustness and stability of approximate reasoning schemas is presented, e.g., in [10,9], where these notions have been considered in a rough mereological framework.

9 On the Approximation Quality of First-Order Theories

So far, we have focused on the generation of approximations to relations using local logical theories in approximation transducers and then building approximation trees from these basic building blocks. This immediately raises the interesting issue of viewing the approximate global theory itself as a conceptual unit. We can then ask what the approximation quality of a theory is and whether we can define qualitative or quantitative measures of the theory's approximation quality. If this is possible, then individual theories can be compared and assessed for their approximative value. One application of this measure would be to choose approximative theories for an application domain at the proper level of abstraction or detail, moving across the different levels of abstraction relative to the needs of the application. In this section, we provide a tentative proposal to compare the approximation quality of first-order theories.

9.1 Comparing Approximation Power of semi-Horn Theories

Definition 11. We say that a *theory* $Th_2(R)$ *better approximates a theory* $Th_1(R)$ *relative to a database* B and denote this by $Th_1(R) \leq_B Th_2(R)$ provided that, in database B, we have $R_1^{+} \subseteq R_2^{+}$ and $R_2^{\oplus} \subseteq R_1^{\oplus}$, where for $i = 1, 2$, R_i^{+} and $R_i^{\oplus}$ denote the lower and upper approximation of R defined by theory Th_i.■

Observe that the notion of a better approximation has a correspondence to information orderings used in the model theory of a number of three-valued and partial logics.

Example 3. Let $CL(x, y)$ denote that objects x, y are close to each other, $SL(x, y)$ denote that x, y are on the same lane, $CH(x, y)$ denote that objects x, y can hit each other, and let $HR(x, y)$ denote that the relative speed of x and y is high. We assume that the lower and upper approximations of these relations can be extracted from data during learning acquisition or are already defined in a database, B. Consider the following two theories approximating the concept $D(x, y)$ which denotes a dangerous situation caused by objects x and y:

- $Th_1(D; CL, SL, CH)$ has two rules:

$$\begin{aligned} &\forall x, y.[(CL(x, y) \wedge SL(x, y)) \to D(x, y)] \\ &\forall x, y.[D(x, y) \to CH(x, y)] \end{aligned} \tag{21}$$

- $Th_2(D; CL, SL, HR)$ has two rules:

$$\begin{aligned} &\forall x, y.[CL(x, y) \to D(x, y)] \\ &\forall x, y.[D(x, y) \to (HR(x, y) \wedge SL(x, y))]. \end{aligned} \tag{22}$$

Using Lemma 1, we can compute the following definitions of approximations of D:

- relative to theory $Th_1(D; CL, SL, CH)$:

$$\begin{aligned} &\forall x, y.[{D^{(1)}}^{+}(x, y) \equiv (CL^{+}(x, y) \wedge SL^{+}(x, y))] \\ &\forall x, y.[{D^{(1)}}^{\oplus}(x, y) \equiv CH^{\oplus}(x, y)] \end{aligned} \tag{23}$$

- relative to theory $Th_2(D; CL, SL, HR)$:

$$\begin{aligned} &\forall x, y.[{D^{(2)}}^{+}(x, y) \equiv CL^{+}(x, y)] \\ &\forall x, y.[{D^{(2)}}^{\oplus} \equiv (HR^{\oplus}(x, y) \wedge SL^{\oplus}(x, y))]. \end{aligned} \tag{24}$$

Obviously ${D^{(1)}}^{+} \subseteq {D^{(2)}}^{+}$. If we additionally assume that in our domain of discourse (and by implication in database B) that $HR \cap SL \subseteq CH$ applies, we can also obtain the additional relation that ${D^{(2)}}^{\oplus} \subseteq {D^{(1)}}^{\oplus}$. Thus $Th_1 \leq_B Th_2$, which means that an agent possessing the knowledge implicit in Th_2 is better able to approximate concept D than an agent possessing knowledge implicit in Th_1.■

These types of comparative relations between theories should prove to be very useful in cooperative agent architectures, but we leave this application for future work.

10 Conclusions and Related Work

In this chapter, we have presented a framework for both the generation, structuring and reasoning about approximate relations having dependencies with each other. We began with a discussion of the the subclass of approximate primitive concepts grounded in sensor or other data via the use of learning techniques. We then introduced the idea of an approximation transducer as a basic constituent in the construction of more complex approximation trees consisting of combinations of a number of approximation transducers. An approximation transducer defines an approximate relation in terms of other approximate relations and a local transducer theory where dependencies between the relations are represented as logical formulas in a traditional manner. This combination of both approximate and crisp knowledge brings together techniques and concepts from two research disciplines. By providing syntactic characterizations of these ideas and techniques, we are able to propose a novel type of approximate knowledge structure which is both elaboration tolerant, elastic, modular and grounded in the particular contexts associated with various applications.

By restricting the syntax of local transducer theories, we can implement the approximation tree inference mechanism in an efficient manner by using a slight generalization of deductive relational databases to include rough relations. Efficient reasoning mechanisms are important because experimentation is being done within the constraints of the WITAS UAV project where these techniques are intended to be used on-board the UAV as an integral part of its knowledge representation mechanisms. In the chapter we used a number of examples specific to the UAV domain to demonstrate the use and versatility of the techniques.

A richer and more complex type of generalized deductive database is proposed in a companion chapter in this volume [2]. In this case, an open-world assumption is assumed about relational information, rather than the standard closed-world assumption view. In addition, the query language used is logical in nature and permits what we call *contextually closed queries*. These queries include the query itself, a local context in the form of integrity constraints and a minimization/maximization policy which permits locally closing parts of the database relative to a query. The latter technique replaces the global closed world assumption and provides finer-grained closure policies which are very useful for open-world planning applications. The use of approximate relations, contextually closed queries and a logical query language imply the use of a special inference mechanism. These generalized deductive databases which we call *rough knowledge databases* provide us with an even richer form of query/answering system which subsumes the idea of approximation trees and can also be used for many other applications involving queries on partial models.[6].

Acknowledgments

The research has been supported in part by the Wallenberg Foundation. Witold Łukaszewicz, Andrzej Skowron and Andrzej Szałas have also been supported by the KBN grant 8 T11C 009 19 and the KBN grant 8T11C 025 19. We thank Jonas Kvarnström for a thorough and valuable proof reading of this chapter.

References

1. P. Doherty, G. Granlund, K. Kuchcinski, K. Nordberg, E. Sandewall, E. Skarman, and J. Wiklund. The WITAS unmanned aerial vehicle project. In *Proceedings of the 14th European Conference on Artificial Intelligence*, pages 747–755, 2000.
2. P. Doherty, J. Kachniarz, and A. Szałas. Using contextually closed queries for local closed-world reasoning in rough knowledge databases. In L. Polkowski and A. Skowron, editors, *Rough-Neuro Computing: Techniques for Computing with Words*, Artificial Intelligence Book Series. Springer-Verlag, 2002. This volume.
3. P. Doherty, W. Łukaszewicz, and A. Szałas. Computing circumscription revisited. *Journal of Automated Reasoning*, 18(3):297–336, 1997. See also: Report LiTH-IDA-R-94-42, Linkoping University, 1994 and Proceedings of the 14th International Joint Conference on AI (IJCAI'95), Morgan Kaufmann Pub. Inc.
4. P. Doherty, W. Łukaszewicz, and A. Szałas. Declarative PTIME queries for relational databases using quantifier elimination. *Journal of Logic and Computation*, 9(5):739–761, 1999. See also: Report LiTH-IDA-R-96-34, Linkoping University, 1996.
5. H-D. Ebbinghaus and J. Flum. *Finite Model Theory.* Springer-Verlag, Heidelberg, 1995.
6. J. McCarthy. Approximate objects and approximate theories. In A.G. Cohn, F. Giunchiglia, and B. Selman, editors, *Proc. 7th International Conf. on Principles of Knowledge Representation and Reasoning, KR2000*, pages 519–526, San Francisco, Ca., 2000. Morgan Kaufmann Pub., Inc.
7. M. Minsky. *The Society of Mind.* Simon & Schuster, Inc., 1986.
8. A. Nonnengart and A. Szałas. A fixpoint approach to second-order quantifier elimination with applications to correspondence theory. In E. Orłowska, editor, *Logic at Work: Essays Dedicated to the Memory of Helena Rasiowa*, volume 24 of *Studies in Fuzziness and Soft Computing*, pages 307–328. Springer Physica-Verlag, 1998. See also: Report MPI-I-95-2-007, Max-Planck-Institut fuer Informatik, Saarbruecken, 1995.

---

[6] For a detailed presentation of rough knowledge databases and the use of contextually closed queries, we refer the reader to the companion chapter in this volume [2].

9. L. Polkowski and A. Skowron. Rough mereology: A new paradigm for approximate reasoning. *International Journal of Approximate Reasoning*, 15(4):333–365, 1996.
10. L. Polkowski and A. Skowron. Towards adaptive calculus of granules. In L.A. Zadeh and J. Kacprzyk, editors, *Computing with Words in Information/Intelligent Systems*, pages 201–227, Heidelberg, 1999. Physica-Verlag.
11. W. V. O. Quine. *The Web of Belief*. McGraw-Hill, 2nd edition edition, 1978.
12. P. Stone. *Layered Learning in Multiagent Systems: A Winning Approach to Robotic Soccer*. MIT Press, 2000.